

Understanding Consistency Maintenance in Service Discovery Architectures during Communication Failure

Chris Dabrowski, Kevin Mills, Jesse Elder

WOSP 2002

Rome, Italy



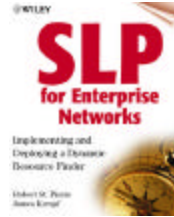



July 25, 2002

Dynamic discovery protocols in essence...

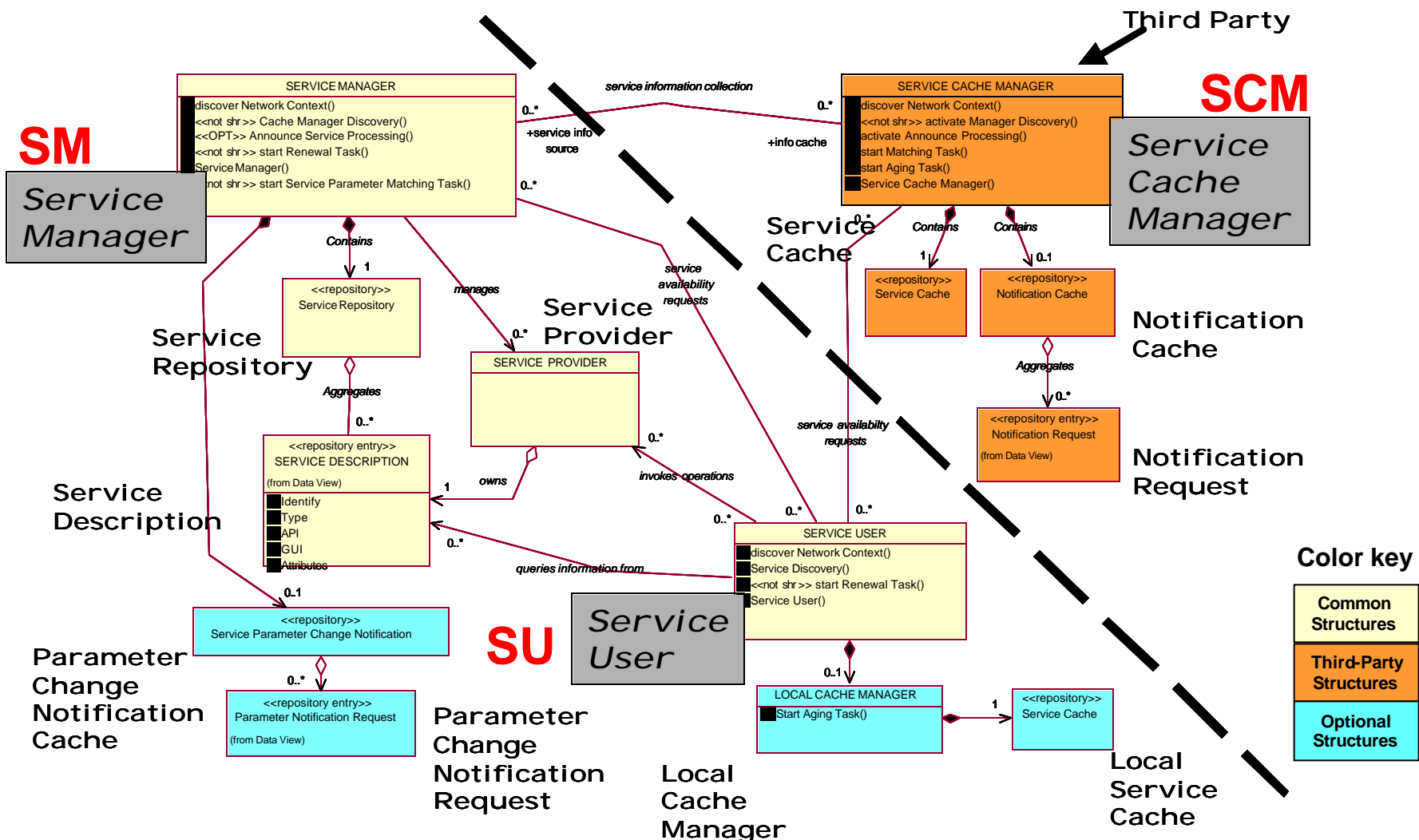
enable ***distributed software components***

- (1) to ***discover*** each other without prior arrangement,
- (2) to ***express*** opportunities for collaboration,
- (3) to ***compose*** themselves into larger collections that cooperate to meet an application need, and
- (4) to ***detect and adapt*** to failures.

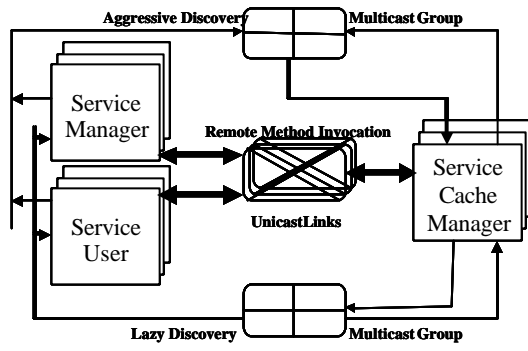
Some examples:

 3-Party Design	 2-Party Design	 Adaptive 2/3-Party Design
 Vertically Integrated 3-Party Design	 Network- Dependent 3-Party Design	 Bluetooth™ Network-Dependent 2-Party Design

General Architecture for Service Discovery Systems



Modeling and Analysis Approach



Behavior Model

Topology

Scenario

Time	Command	Parameters
5	NodeFail	SM4
5	LinkFail	SCM1 SM4
10	GroupJoin	SM4 GROUP1
10	FindService	SU8 5 1 2 S XYZ ALL
50	AddService	SM4 SCM3 T ATT API GUI 20 30



Execute with Rapide

Consistency Conditions

- For All (SM, SD, SCM):
 - (SM, SD) IsElementOf SCM registered -services (CC1)
 - implies SCM IsElementOf SM discovered -SCMs
- For All (SM, SD, SCM):
 - SCM IsElementOf SM discovered -SCMs & (SD) IsElementOf SM managed -services (CC2)
 - implies (SM, SD) IsElementOf SCM registered -services
- For All (SM, SD, SCM):
 - SCM IsElementOf SM discovered -SCMs & (SM, SD) IsElementOf SCM registered -services & NOT (SCM IsElementOf SM persistent -list) (CC3)
 - implies Intersection (SM GroupsToJoin, SCM GroupsMemberOf)
- For All (SM, SD, SCM, SU, NR):
 - (SU, NR) IsElementOf SCM requested -notifications & (SM, SD) IsElementOf SCM registered -services & Matches((SM, SD), (SU, NR)) (CC4)
 - implies (SM, SD) IsElementOf SU matched -services



Analyze POSETs

Use metrics to Assess Correctness & Performance

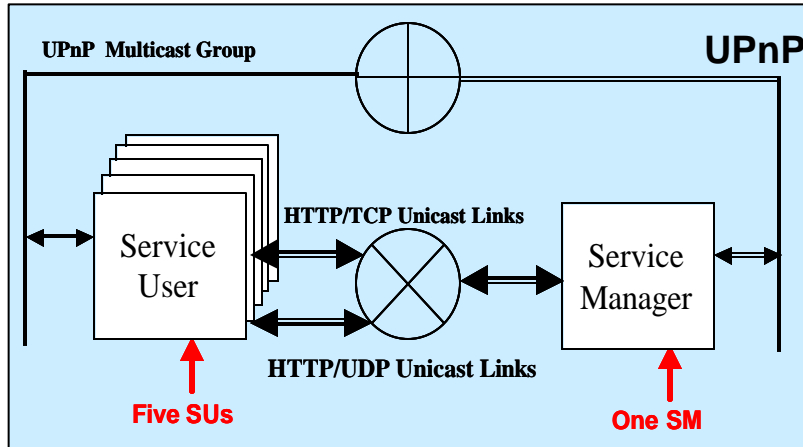
```

-- *****
-- ** 3.3 DIRECTED DISCOVERY CLIENT INTERFACE **
-- *****
-- This is used by all JINI entities in directed
-- discovery mode. It is part of the SCM_Discovery
-- Module. Sends Unicast messages to SCMs on list of
-- SCMs to be discovered until all SCMs are found.
-- Receives updates from SCM DB of discovered SCMs ah
-- removes SCMs accordingly
-- NOTE: Failure and recovery behavior are not
-- yet defined and need review.
TYPE Directed_Discovery_Client
(SourceID : IP_Address; InSCMsToDiscover : SCMList; StartOption : DD_Code;
 InRequestInterval : TimeUnit; dMaxNumTries : integer; InPV : ProtocolVersion)
IS INTERFACE
SERVICE DDC_SEND_DIR : DIRECTED_2_STEP_PROTOCOL;
SERVICE DISC_MODES : dual SCM_DISCOVERY_MODES;
SERVICE DD_SCM_Update : DD_SCM_Update;
SERVICE SCM_Update : SCM_Update;
SERVICE DB_Update : dual DB_Update;
SERVICE NODE_FAILURES : dual NODE_FAILURES; -- e vents for failure and recovery.
ACTION
IN Send_Requests(),
BeginDirectedDiscovery();
BEHAVIOR
action animation_lam(name: string);
MySourceID : VAR IP_Address;
PV : VAR ProtocolVersion;

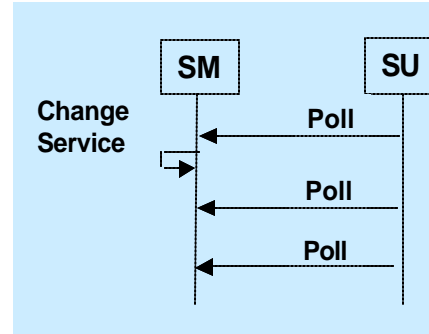
```

7/25/02

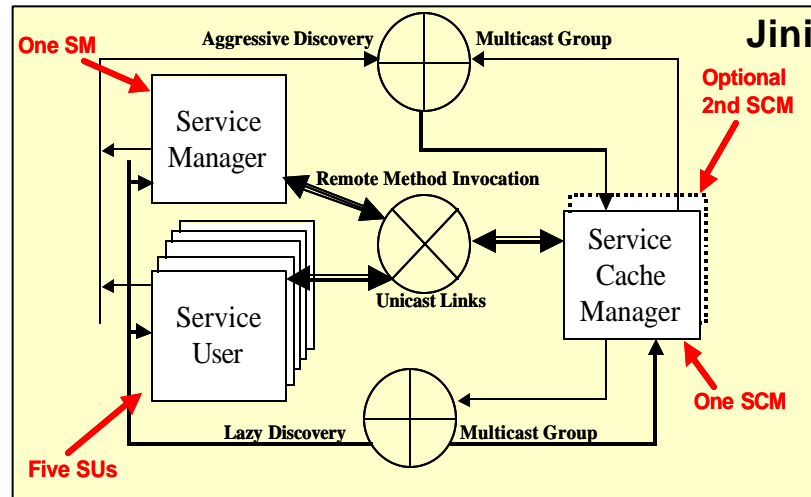
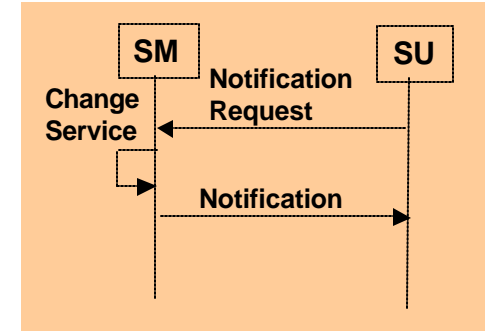
How do various service discovery architectures, topologies, and consistency-maintenance mechanisms perform under deadline during communication failure?



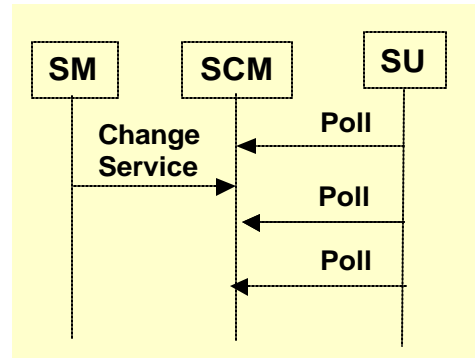
Two-Party Polling



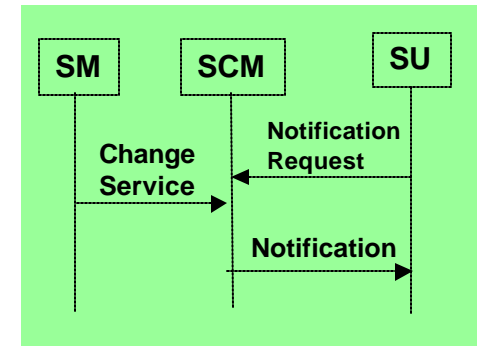
Two-Party Notification



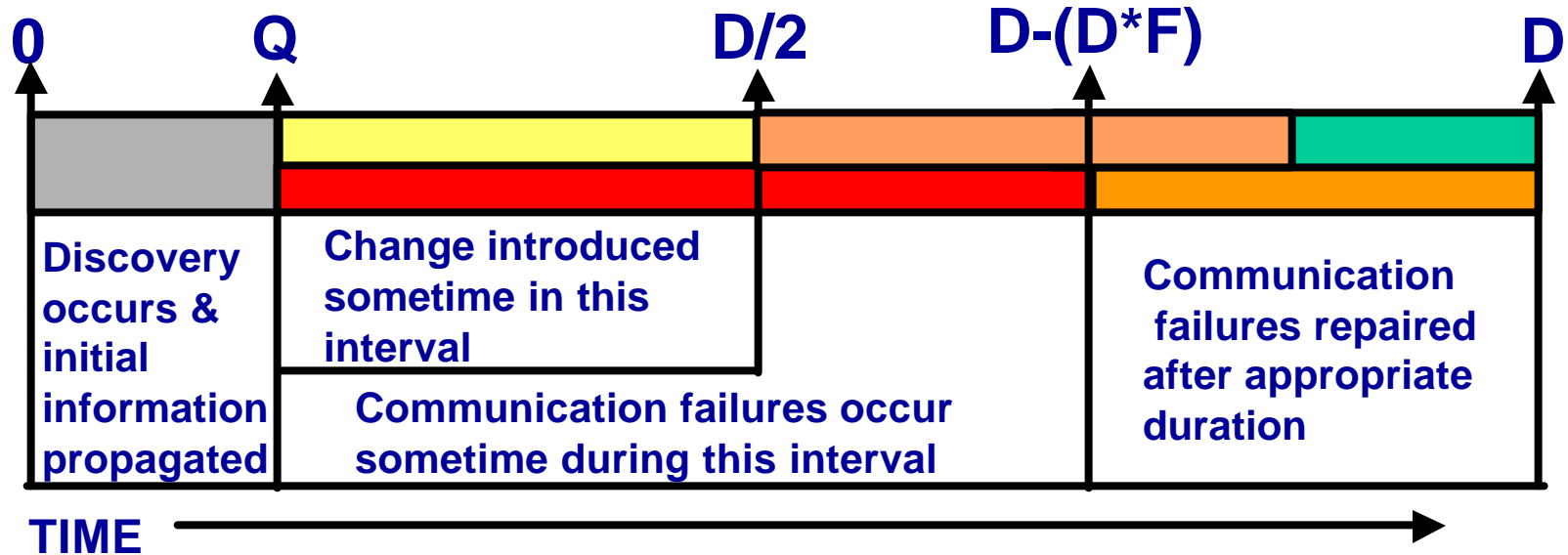
Three-Party Polling



Three-Party Notification



Modeling Communication Failures



Random Processes

1. Choose a time to introduce the change [uniform(Q, D/2)]
2. For each node, choose a time to introduce a communication failure [uniform(Q, D-(D*F))]
3. When each failure occurs, choose a scope for the failure, where each of [Rx, Tx, Both] has an equal probability

Q = end of quiescent period (100 s in our experiment)

D = propagation deadline (5400 s in our experiment)

F = failure duration (variable from 0% - 75% in 5% increments in our experiment)

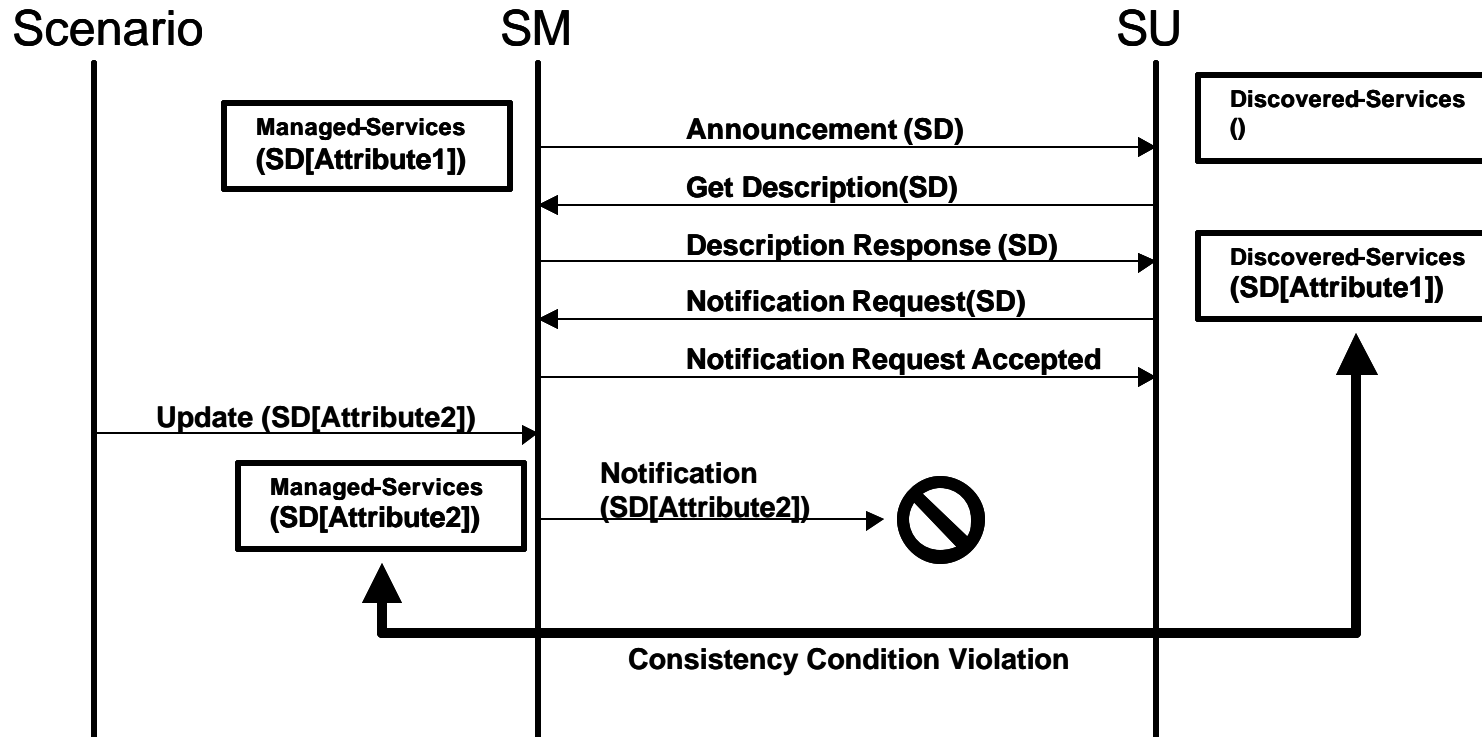
Monitoring Consistency

For All (SM, SU, SD):

(SM, SD [Attributes1]) *IsElementOf* SU discovered-services

SD [Attributes2] *IsElementOf* SM managed-services

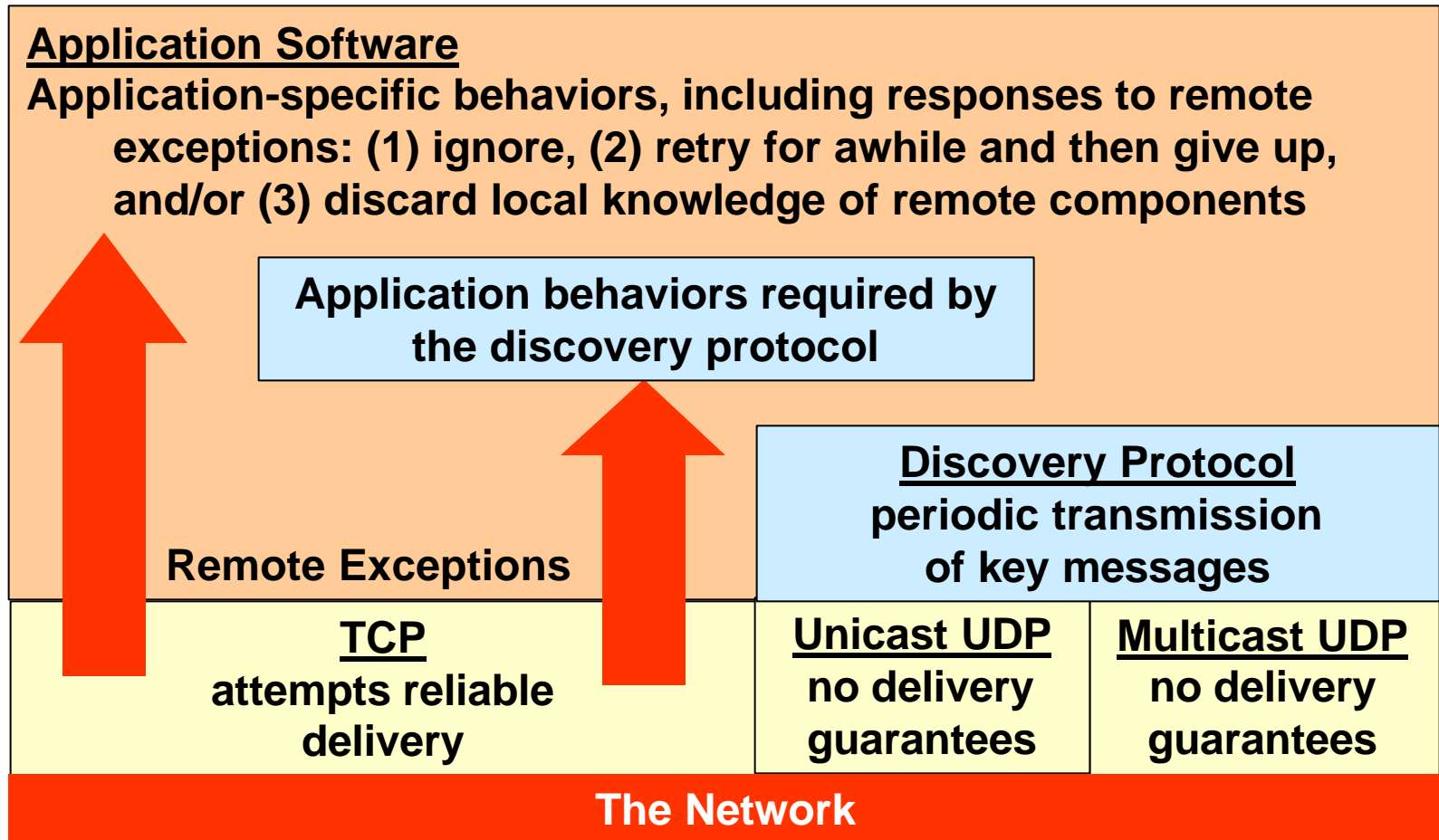
***implies* Attributes1 = Attributes2**



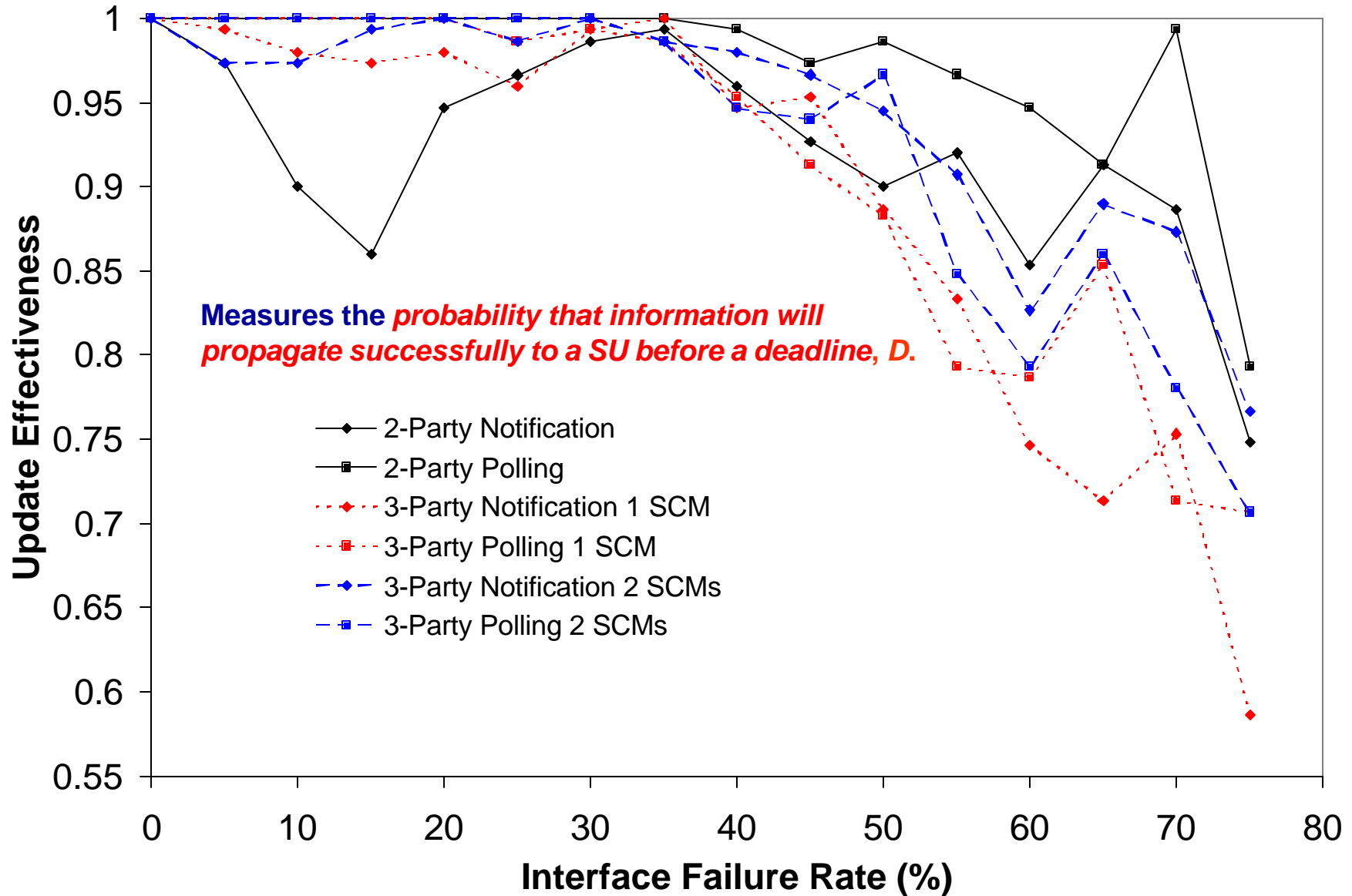
How well does the system restore consistency after restoration of communication?

Division of Failure Recovery Responsibilities:

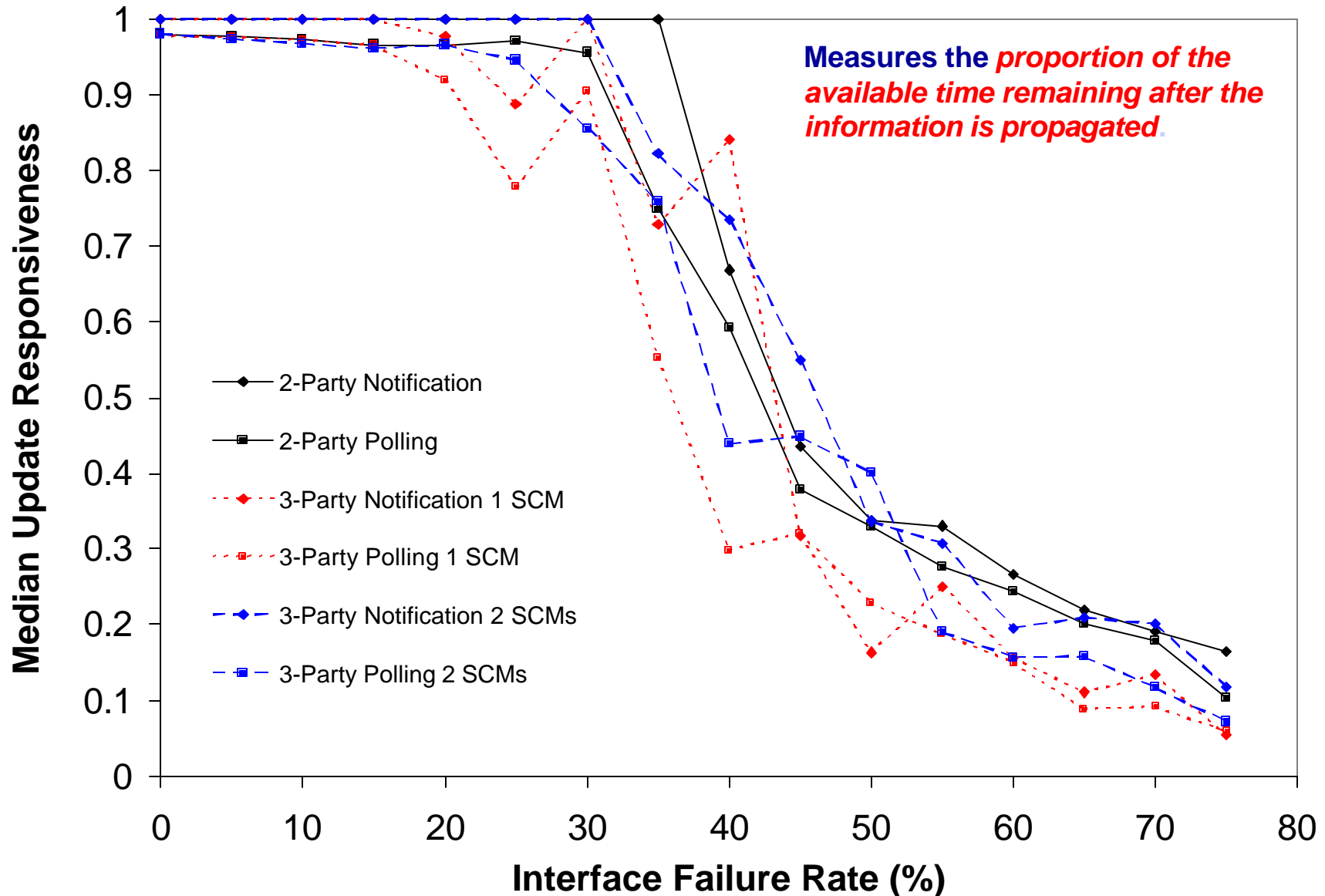
Communication Protocol - *Discovery Protocol* - *Application Software*



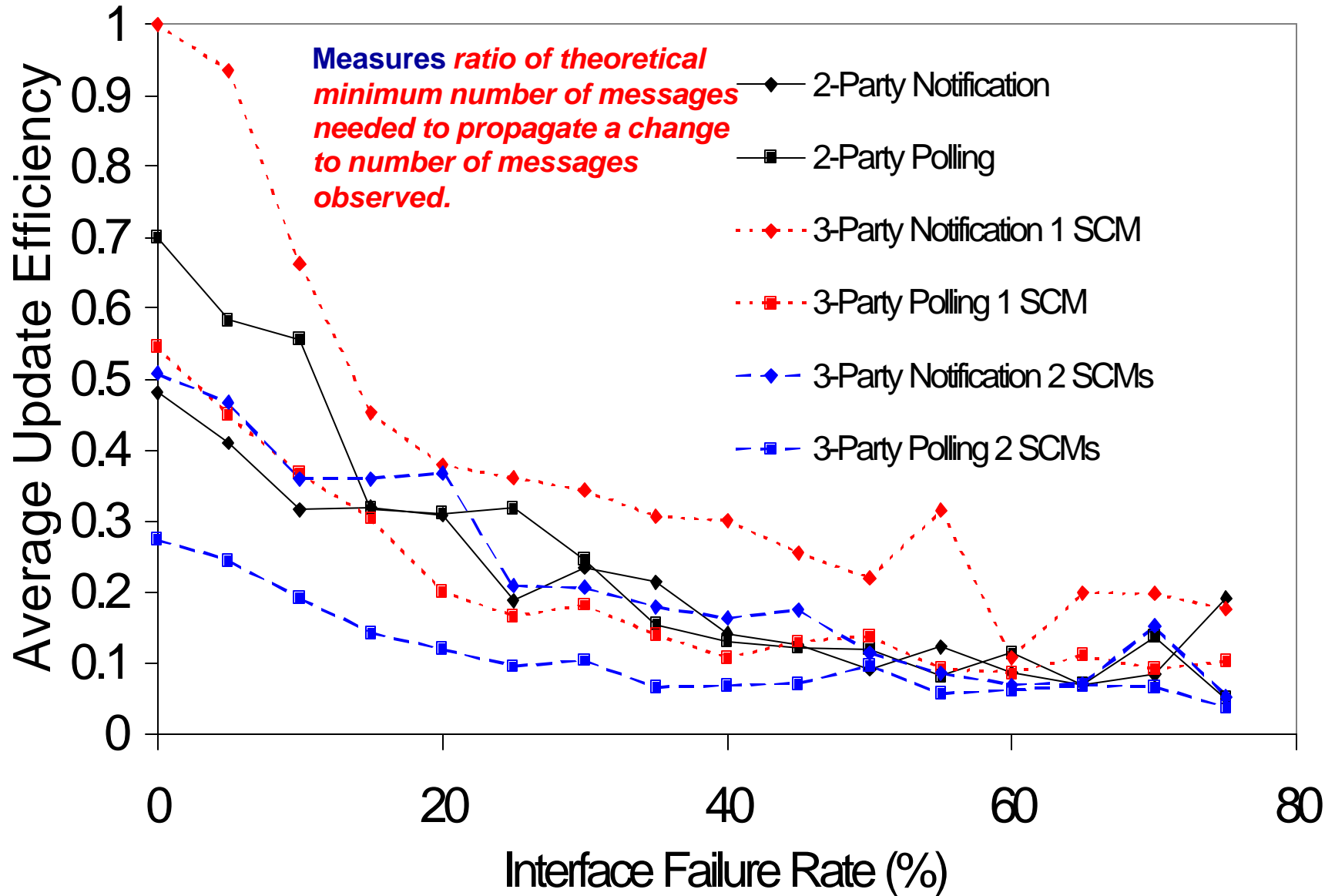
Update Effectiveness UPnP (2-Party) vs. Jini (3-Party)



Update Responsiveness UPnP (2-Party) vs. Jini (3-Party)



Update Efficiency UPnP (2-Party) vs. Jini (3-Party)



Conclusions

- **Executable architectural models represent essential complexity and reveal collective dynamics – leading to valuable insights**
 - paper specifications do not represent dynamics very well
 - reference implementations exhibit substantial incidental complexity
- **A single architectural model can be analyzed for behavioral, performance, and logical properties**
 - limits errors and inconsistencies that can creep in when using multiple models to represent different facets of a design
- **2-party and 3-party discovery architectures share similar robustness properties during communication failure, but**
 - sole reliance on TCP retransmissions to recover notifications leads to an unexpected saw-tooth in update effectiveness, which is most pronounced for UPnP (Jini includes some SM behaviors which compensate)
 - adding a redundant SCM in the 3-party architecture improves effectiveness and responsiveness nearly to the level of the 2-party architecture, but adding a redundant SCM also lowers efficiency